

# **Отчёт по лабораторной работе №10**

**Дисциплина : архитектура компьютера**

**Зарина Исмайлбековна Исаева**

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
4.1	Написание программ для работы с файлами . . . . .	9
4.2	Задание для самостоятельной работы . . . . .	11
<b>5</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

## **Список таблиц**

## Список иллюстраций

4.1	Создание файлов для лабораторной работы . . . . .	9
4.2	Ввод текста программы из листинга 10.1 . . . . .	10
4.3	Запуск программного кода . . . . .	10
4.4	Запрет на выполнение файла . . . . .	10
4.5	Добавление прав на использование . . . . .	11
4.6	Предоставление прав доступа в символьном и двоичном виде . .	11
4.7	Написание текста программы . . . . .	12
4.8	Запуск исполняемого файла и проверка его работы . . . . .	12

# **1 Цель работы**

Приобретение навыков написания программ для работы с файлами.

## **2 Задание**

1. Написание программ для работы с файлами.
2. Задание для самостоятельной работы.

### 3 Теоретическое введение

Права доступа определяют набор действий (чтение, запись, выполнение), разрешённых для выполнения пользователям системы над файлами. Для каждого файла пользователь может входить в одну из трех групп: владелец, член группы владельца, все остальные. Для каждой из этих групп может быть установлен свой набор прав доступа.

Для изменения прав доступа служит команда `chmod`, которая понимает как символьное, так и числовое указание прав.

Обработка файлов в операционной системе Linux осуществляется за счет использования определенных системных вызовов. Для корректной работы и доступа к файлу при его открытии или создании, файлу присваивается уникальный номер (16-битное целое число) – дескриптор файла.

Для создания и открытия файла служит системный вызов `sys_creat`, который использует следующие аргументы: права доступа к файлу в регистре `ECX`, имя файла в `EBX` и номер системного вызова `sys_creat` (8) в `EAX`.

Для открытия существующего файла служит системный вызов `sys_open`, который использует следующие аргументы: права доступа к файлу в регистре `EDX`, режим доступа к файлу в регистр `ECX`, имя файла в `EBX` и номер системного вызова `sys_open` (5) в `EAX`.

Для записи в файл служит системный вызов `sys_write`, который использует следующие аргументы: количество байтов для записи в регистре `EDX`, строку содержимого для записи `ECX`, файловый дескриптор в `EBX` и номер системного вызова `sys_write` (4) в `EAX`. Системный вызов возвращает фактическое количество

записанных байтов в регистр EAX. В случае ошибки, код ошибки также будет находиться в регистре EAX. Прежде чем записывать в файл, его необходимо создать или открыть, что позволит получить дескриптор файла.

Для чтения данных из файла служит системный вызов `sys_read`, который использует следующие аргументы: количество байтов для чтения в регистре EDI, адрес в памяти для записи прочитанных данных в ECX, файловый дескриптор в EBX и номер системного вызова `sys_read` (3) в EAX. Как и для записи, прежде чем читать из файла, его необходимо открыть, что позволит получить дескриптор файла.

Для правильного закрытия файла служит системный вызов `sys_close`, который использует один аргумент – дескриптор файла в регистре EBX. После вызова ядра происходит удаление дескриптора файла, а в случае ошибки, системный вызов возвращает код ошибки в регистр EAX.

Для изменения содержимого файла служит системный вызов `sys_lseek`, который использует следующие аргументы: исходная позиция для смещения EDI, значение смещения в байтах в ECX, файловый дескриптор в EBX и номер системного вызова `sys_lseek` (19) в EAX. Значение смещения можно задавать в байтах.

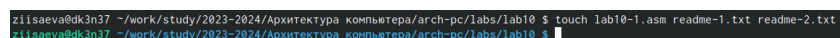
Удаление файла осуществляется системным вызовом `sys_unlink`, который использует один аргумент – имя файла в регистре EBX.



## 4 Выполнение лабораторной работы

### 4.1 Написание программ для работы с файлами

Создаю каталог для программ лабораторной работы № 10, перехожу в него и создаю файлы lab10-1.asm, readme-1.txt и readme-2.txt (рис. [4.1]).



```
zllsaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ touch lab10-1.asm readme-1.txt readme-2.txt
zllsaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $
```

Рис. 4.1: Создание файлов для лабораторной работы

Ввожу в файл lab10-1.asm текст программы, записывающей в файл сообщения, из листинга 10.1 (рис. [4.2]).

```

/afs/.dk.sci.pfu.edu.ru/home/z/i/ziisaeva/work/study/202
;-----
; Запись в файл строки введенной на запрос
;-----
%include 'in_out.asm'
SECTION .data
filename db 'readme.txt', 0h ; Имя файла
msg db 'Введите строку для записи в файл: ', 0h ; Сообщение
SECTION .bss
contents resb 255 ; переменная для вводимой строки
SECTION .text
global _start
_start:
; --- Печать сообщения 'msg'
mov eax,msg
call sprint
; ---- Запись введенной с клавиатуры строки в 'contents'
mov ecx, contents
mov edx, 255
call sread
; --- Открытие существующего файла ('sys_open')
mov ecx, 2 ; открываем для записи (2)
mov ebx, filename
mov eax, 5
int 80h

```

Рис. 4.2: Ввод текста программы из листинга 10.1

Создаю исполняемый файл и запускаю его (рис. [4.3]).

```

ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ nasm -f elf lab10-1.asm
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ld -m elf_i386 -o lab10-1 lab10-1.o
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ./lab10-1
Введите строку для записи в файл: pied piper
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ cat readme-1.txt

```

Рис. 4.3: Запуск программного кода

Далее с помощью команды `chmod u-x` изменяю права доступа к исполняемому файлу `lab10-1`, запретив его выполнение и пытаюсь выполнить файл (рис. [4.4])

```

ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ chmod u-x lab10-1
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ./lab10-1
bash: ./lab10-1: Отказано в доступе
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $

```

Рис. 4.4: Запрет на выполнение файла

Файл не выполняется, т.к в команде я указала “u” - владелец (себя), “-” - отменить набор прав, “x” - право на исполнение.

С помощью команды `chmod u+x` изменяю права доступа к файлу `lab10-1.asm` исходным текстом программы, добавив права на исполнение, и пытаюсь выполнить его (рис. [4.5])

```
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ chmod u+x lab10-1.asm
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ./lab10-1.asm
./lab10-1.asm: строка 1: синтаксическая ошибка рядом с неожиданным маркером «;»
./lab10-1.asm: строка 1: `;-----'
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $
```

Рис. 4.5: Добавление прав на использование

Текстовый файл начинает исполнение, но не исполняется, т.к не содержит в себе команд для терминала.

В соответствии со своим вариантом (3) в таблице 10.4 предоставляю права доступа к файлу `readme1.txt` представленные в символьном виде, а для файла `readme-2.txt` – в двоичном виде:

`r-x -wx rw-, 011 101 011`

И проверяю правильность выполнения с помощью команды `ls -l` (рис. [4.6]).

```
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ chmod 640 readme-1.txt # r-x -wx rw-
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ chmod 640 readme-2.txt # 011 101 011
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ls -l
итого 21
-rw-r--r-- 1 ziisaeva studsci 3942 ноя 27 15:39 in_out.asm
-rwxr-xr-x 1 ziisaeva studsci 9164 ноя 27 15:48 lab10-1
-rwxr--r-- 1 ziisaeva studsci 1287 ноя 27 15:48 lab10-1.asm
-rw-r--r-- 1 ziisaeva studsci 1472 ноя 27 15:48 lab10-1.o
drwxr-xr-x 3 ziisaeva studsci 2048 сен 25 14:13 presentation
-rw-r----- 1 ziisaeva studsci  0 ноя 27 15:37 readme-1.txt
-rw-r----- 1 ziisaeva studsci  0 ноя 27 15:37 readme-2.txt
drwxr-xr-x 5 ziisaeva studsci 2048 сен 25 14:13 report
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $
```

Рис. 4.6: Предоставление прав доступа в символьном и двоичном виде

## 4.2 Задание для самостоятельной работы

Пишу код программы, выводящей приглашения “Как Вас зовут?”, считывающей с клавиатуры фамилию и имя и создающую файл, в который записывается сообщение “Меня зовут”ФИ”” (рис. [4.7])

```

GNU nano 7.2 /afs/.dk.sci.pfu.edu.ru/home/z/i/ziisaeva.
%include 'in_out.asm'
SECTION .data
msg1 db 'Как Вас зовут?', 0h
filename db 'name.txt', 0h
msg2 db 'Меня зовут ', 0h
SECTION .bss
name resb 255
SECTION .text
global _start
_start:
mov eax,msg1
call sprintLF
mov ecx, name
mov edx, 255
call sread
mov ecx, 0777o
mov ebx, filename
mov eax, 8
int 80h
mov ecx, 2
mov ebx, filename
mov eax, 5
int 80h
mov esi, eax

```

Рис. 4.7: Написание текста программы

Создаю исполняемый файл и проверяю его работу. Проверяю наличие файла и его содержимое с помощью команд `ls` и `cat` (рис. [4.8]).

```

ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ touch ex.asm
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ nasm -f elf ex.asm
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ld -m elf_i386 -o ex ex.o
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ls
ex ex.asm ex.o in_out.asm lab10-1 lab10-1.asm lab10-1.o presentation readme-1.txt readme-2.txt report
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ./ex
Как Вас зовут?
Исаева Зарина
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ ls
ex ex.asm ex.o in_out.asm lab10-1 lab10-1.asm lab10-1.o name.txt presentation readme-1.txt readme-2.txt report
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $ cat name.txt
Меня зовут Исаева Зарина
ziisaeva@dk3n37 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab10 $

```

Рис. 4.8: Запуск исполняемого файла и проверка его работы

Код программы:

```

%include 'in_out.asm'
SECTION .data
msg1 db 'Как Вас зовут?', 0h
filename db 'name.txt', 0h
msg2 db 'Меня зовут ', 0h
SECTION .bss
name resb 255
SECTION .text
global _start
_start:
mov eax,msg1
call sprintLF
mov ecx, name
mov edx, 255
call sread

```

```
mov ecx, 0777o mov ebx, filename mov eax, 8 int 80h mov ecx, 2 mov ebx, filename  
mov eax, 5 int 80h mov esi, eax mov eax, msg2 call slen mov edx, eax mov ecx, msg2  
mov ebx, esi mov eax, 4 int 80h mov eax, name call slen mov edx, eax mov ecx, name  
mov ebx, esi mov eax, 4 int 80h mov ebx, esi mov eax, 6 int 80h call quit
```

## **5 Выводы**

Благодаря данной лабораторной работе я приобрела навыки написания программ для работы с файлами.

## Список литературы

1. [Лабораторная работа №10] ([https://esystem.rudn.ru/pluginfile.php/2089097/mod\\_resource/c](https://esystem.rudn.ru/pluginfile.php/2089097/mod_resource/c))